Three factor authentication system with modified ECC based secured data transfer: untrusted cloud environment

Dilip Venkata Kumar Vengala¹ · D. Kavitha² · A. P. Siva Kumar³

Received: 25 November 2020 / Accepted: 16 February 2021
© The Author(s) 2021

Abstract

Cloud computing (CC) is a technology that delivers its service by means of the internet. In the modern scenario, cloud storage services have gained attention. The cloud environment confronts data breaches expansively in cloud storage, which might bring about the disclosure of personal in addition to corporate data. Thus, the requirement arises for the creation of a more foremost authentication system. Customary authentication schemes depend on techniques, like Password Authentications Protocol (PAP), Challenge Handshakes Authentication Protocols (CHAP), as well as One-Time Pads (OTP), which are often susceptible to malevolent attacks as well as security threats. To shun such issues, this paper proposed a Modified ECC centred secure data transfer and a '3'-factor authentication scheme in the untrusted cloud environment. The proposed work comprises '3' steps: authentication, data compression, and safe data transfer. In the authentication phase, the SHA-512 algorithm along with CCP is utilized. After that, the user-uploaded data is compressed utilizing CHA on the server-side. Next, MECC encrypts the compressed data, and then, safely uploaded it to the cloud server (CS). In the investigational appraisal, the proposed work is contrasted with the prevailing methods. The outcomes proved that the proposed work renders better security than the prevailing methods.

Keywords Compliments hashing algorithm (CHA) · Modified Elliptic Curve Cryptographic (MECC) · Cued Click Point (CCP) · Three-factor authentication

Introduction

CC has attained significance in the present situation [1]. It is presented as on-demand computing, software-as-a-service (SaaS), or the Internet as a platform. The fundamental process is the sweeping of data along with programs as of desktop PC, corporate server quarters and also installed in "the compute cloud" [2]. The exposé of CC has impacted an

enormous effect on the Information Technology (IT) along with other industries for the precedent some decades [3], wherein a lot of business similar to Google, Amazon, along with Microsoft endeavours to offer consistent along with cost-effective cloud platforms [1]. The '4' employed design of CC are public cloud, private cloud, hybrid cloud along with virtual private cloud [4, 5], in addition, the '3' service designs are, Infrastructure-as-a-services (IaaS), Platforms-as-a-Services (PaaS), in conjunction with SaaS [6]. Even though the clouds offer better storage, it lacks security. The '2' main issues of CC are security along with privacy since clients encompass very little access to their stored data at the isolated locations administered by disparate service suppliers [7].

The government along with public sectors have given foremost importance to Internet security to a degree that it has turned into a separate division in CC [8]. The internet security that is really capable can protect the organization's finances along with their other confidential data, whereas the incapable security can lead the data to be susceptible to attack, which consecutively might crumple a system.

✉ Dilip Venkata Kumar Vengala
dilipvenkatakumar@gmail.com

D. Kavitha
dwaramkavithareddy@gmail.com

A. P. Siva Kumar
sivakumar.ap@gmail.com

¹ Department of Computer Science and Engineering, JNTUA, Anantapur, India

² Department of Computer Science and Engineering, G.Pulla Reddy Engineering College, Kurnool, India

³ Department of Computer Science and Engineering, JNTUA College of Engineering, Anantapur, India

Security includes ‘3’ main features: confidentiality, integrity, availabilitys (CIA). For maximum protection, these aspects are given more importance whilst constructing the security measure [9]. Internet security depends on particular resources along with standards for defending the data that is transmitted via internet viz., encryption, firewalls [10], which obstruct unnecessary traffic, anti-malware, anti-spyware together with anti-virus programs.

The data comprises a life cycle of six steps: (1) create, (2) store, (3) use, (4) share, (5) archive and also (6) destroy. The defense of data in every one of these six steps is necessary. The privacy of the data whilst sharing by means of the internet is made certain via numerous access control mechanisms [11], say PAP, CHAP, extensible authentication protocol together with cryptographic methods. PAP is susceptible to being read as of the point-to-point protocol (PPP) data packets swapped betwixt the authentication server and the consumer’s machine. PAP is utilized in association with OTP for improved security [12]. The cryptographic encryption algorithms are generally be bifurcated into symmetric keys encryption algorithms as well as asymmetric keys encryption algorithms [13]. The degree of complexity of the access control machinery ought to be comparative to the worth of the information being secured.

Until it is moved to the cloud for storage, cloud service providers provide cloud strong encryption to encrypt information. Classic implementations for cloud encryption vary from encrypted connections to restricted encryption of only confidential data to end-to-end encryption of all data submitted to the cloud. The virtualization technology, mainly virtual machines (VM) utilized in CC have raised exclusive security along with survivability threats for cloud users [13]. The cloud data is susceptible to attacks, say Brute force attack, Man-in-middle attack (MITM), as well as a Dictionary attack. CC has many advantages viz., (a) no up-frontal investments necessary, (b) the resources in the cloud are quickly allotted as well as re-allotted, therefore, peak load provisions are not necessary, (c) low operating cost, (d) the cloud services are web-centred, therefore, can be effortlessly accessed and also could offer accessibility to an extensive range of services [10] along with (5) every service is scalable [9]. As of today, numerous individual cloud storage applications exist, like Google Drive, Apple iCloud, Microsoft OneDrive, as well as Dropbox. Nevertheless, Dropbox is the utmost well-liked one and trounces the others with respect to users and formed traffic [14]. Accessing Dropbox directly will bring about attacks, which cause the disclosure of the user data. Accordingly, there is a requirement for an effective authentication method for safe data transmission on cloud infrastructure.

In current times, numerous authentication methods, say integrated key cryptography (IKC), encryption algorithms, PAP, etc. are established for safe data transmission in the

cloud. Nevertheless, those methods are still defenseless to several malware attacks along with security threats. Thus, this paper introduced a technique for data security in Cloud Server via the MECC along with three-factor authentication.

The remaining section of the provided paper is constructed as trails. “[Proposed secure data transfer methodology](#)” surveys the associated work concerning the proposed work. “[Result and discussion](#)” renders a concise discussion of the proposed work. “[Conclusion](#)” analyses the experimental outcomes and deduces the given paper.

Literature survey

Halabi et al. [15] recommended a broker-centered structure that administered the Cloud defense-Service Level Agreement (SLA). A standard, quantitative, along with a measurable form of the agreement was generated originally, which defeated the problems of the remaining security SLA. They illustrated that the suggested technique in a quantified standard SLA enhanced the speed of the cloud adaptation process. After that assessment was performed centered on calculating the sufficient trade-off betwixt the security CIA triad features in the circumstance of a multi-objective optimization issue. This doesn’t render integrity together with security to the anticipated level.

Jeong et al. [16] suggested a multifactor mobile device authentication that rendered security, effectiveness, along with consumer expediency for mobile device usage in cloud service architectures. The security was enhanced by means of strengthening the user’s authentication that was needed prior to accessing cloud computing utilities, and also the authentication keys’ strength was enhanced by means of setting numerous features for authentication. The chief contribution was to improve the security via the mobile devices’ authentication with several features simultaneously and to utilize the mobile cloud service construction effortlessly concerning execution time. This authentication method was significantly affected by means of users’ body conditions together with surrounding environments.

Pei et al. [17] suggested application programming interface (API)-level security certification of android applications (ASCAA), a cloud-centered structure. It engaged the ordered technique to recognize and also examine security concerns at the API level. They also supplied ASCAA security verbal communication, which gave an arrangement to the security rules along with certification procedure. This made the ASCAA scalable. Additionally, the API kernel was organized into cloud surroundings along with that an open interface was offered to every consumer. The ASCAA was established to supply higher performance along with screen threat features with better concentration. The major

drawback of ASCAA was that it allowed some malicious applications, which would result in different security attacks.

Hussain et al. [18] recommended a multilevel classification model of disparate security assaults athwart disparate cloud services at every layer. They also brought the impacts of diverse cloud assaults to the public. The multiple-level classification improved security for every layer on the cloud along with that it determined the kind of security requirements for the server along with the consumer. It supplied an entire disparate system to tackle the security problems as well as to reduce the outcomes. Nevertheless, this method was still suffered to render security in every layer on the cloud chiefly because of high risks.

Kumar et al. [19] proffered the integrated key cryptosystem (IKC) meant for numerous file sharing with a solitary combined key for sole-consumer data sharing. It was the blend of disparate security schemes in attributes-centered encryption. The IKC had two major phases for protecting data sharing. The method was one amongst the public sector cryptosystem utilized to assist the abandonment of secret keys for disparately encrypted files on a cloud storage scheme. The system was contrasted with the other systems; it was extra flexible along with an efficient process for data sharing with a sole aggregate key production on cloud storage. In applications of work with big amounts of encrypted data on a usual base, the IKC would be extremely slow.

Roy and Dasgupta [20] introduced an adaptive multi-factor authentication (MFA) that regarded the effect of disparate consumer devices, media, surroundings, along with the frequency of authentication to perceive the justifiable consumer. Originally, it estimated the impact of the consumer devices, media, environments along with the frequency of authentication. Furthermore, the weight of the set of obtainable modalities on consumer devices was also estimated. Next, the fuzzy sets were definite, after that the adaptive assortment of the authentication modalities completed centered on Sugeno's fuzzy inference [24]. The MFA's security level (SL) doesn't advance to the anticipated level.

Pitchai et al. [21] suggested a searchable encrypted data file sharing scheme (SEDFS). A keyword was allotted for every data throughout encryption. The outcome illustrated augmented efficiency along with reduced search time. The system elucidated the disadvantages of the L-EncDB lightweight encryption in support of the database along with MSSK (most significant single keyword) that made use of the radix sort to investigate the data, along with that the system trounced the augmented storage space centred upon SEDFS [25]. The main shortcoming of SEDFS was that it encompassed more time when searching a keyword.

Rani and Geethakumari [22] rendered three algorithms, say B-tree Huffmans Encoding (BHE), modified elliptic curves cryptography (MECC) as well as deep learning modified neural network (DLMNN) for a safe data transmission

together with detection of Anti-Forensics Attacks on cloud environments. Here, originally the inputted data was compressed in addition to encrypted utilizing BHE and MECC correspondingly [26]. The DLMNN recognized the attacked and non-attacked data. In this, when numerous users asked the cloud to secure data transfer, the network traffic would occur.

Shen et al. [23] offered a method for a multiple-security-level cloud storage system, which was joined with AES symmetric encryption as well as enhanced identity-centred proxy re-encryptions (PRE). The optimization incorporated support intended for fine-grained control in addition to performance optimization. The fine-grained traits meant that the data owner could share confidential data utilizing a fine-grained system, for instance, adding a solo file or a class of files. The PRE suffered numerous security issues; thus, security amelioration was required for this system.

Proposed secure data transfer methodology

Cloud Server is a file hosting service that renders individual spaces for each of its users to hoard their data in the cloud space. With the utilization of the specified username along with the password, the user-space is accessed. Giving direct access to CS will bring about attacks that can expose the user's data. For this reason, an intermediary application is required to be run betwixt the user and the cloud to ensure security. This paper proposed a modified ECC-centred secure data transfer in addition to the three-factor authentication system in the cloud. The proposed work comprises three steps,

- Authentication
 - o Registration
 - o Login
 - o Verification
- Data Compression
 - o Compliments Huffman Algorithm (CHA)
- Secure data transfer
 - o Modified ECC algorithm

The proposed work's block diagram is exhibited in Fig. 1.

Authentication

Authentication is the initial and imperative step in rendering access to approved users. Prior to a user can upload their data in the CS, they need the administration's approval. Subsequent to corroboration, the administrator renders the user

Fig. 1 Block diagram for the proposed methodology

with data for authentication. Next, the users are approved for accessing the system for vital information. This phase contains three steps.

- Registration
- Login
- Verification

Those steps are briefly elucidated as,

(a) Registration

In this, the user inputs the details say name, sex, address, age, password, along with captcha, the system produces an image for the specific user automatically, then the user identifies and clicks five points within the image. These points are along with user profile vector $U(P_v)$ is amassed in the database. Here, the Secure

Hashing Algorithms-512 (SHA-512) transmutes the unique username into the Hashed value. SHA-512 is a hashing algorithm used for Internet authentication, digital signatures, and even bitcoin blockchains. Input formatting will be processed here as blocks of 1024 bits each, so to deal with size and bits, each block should have 1024 bits. Initialization of the Hash buffer has a buffer to store computational value. From each frame, the intermediate results are all used to calculate the next block to achieve the required hash function from our original post.

Lastly, this transmuted Hash value was amassed in the server. The SHA-512 is elucidated as,

Figure 2 exhibits the SHA-512 process, which is safer than the most hashing algorithm, such as Whirlpool, Message Digest (MD5), and RACE Integrity Primitives

Fig. 2 Illustration of SHA-512

Evaluation Message Digest (RIPEMD). The username is set to the 128bit length field. After that, split the augmented data into blocks utilizing a 64bit username that is derived as of the current block, centred on the square root of the initial eight prime numbers (2–19). Next, update the 512-bit buffer utilizing a round constant centred on the cube root of the initial 80 prime numbers (2–409). In this SHA-512, the hashed function is the chief function. It considers the varied-length blocks of data username (U) as input and outputs a fixed-size hashed value. The hashed function is mathematically written as,

$$f_o = H(U) \quad (1)$$

wherein f_o implies the fixed length output, $H(U)$ signifies the hash value of username.

(b) Login

Once the users logged in, they should give the authentication data that is rendered by means of the supervisor for authentication. The user will enter their login information (username, password) and then submit. Then, once the image window is opened, the user picks the same five points on the specific image, and subsequently, the user clicks the submit button. In this

phase, the profile vector is generated. The combination of user name (U), password (P_d), in addition to CCP (C_p) is mathematically written as,

$$P_v = \{U, P_d, C_p\} \quad (2)$$

wherein P_v signifies the profile vector on the login phase.

(c) Verification

Here, the profile vector as of the login phase is contrasted with the ones on the database. When it matches, the login is granted, if not discarded. Centred on three factors: (1) username password, (2) hash code and (3) Cued Click Point (CCP), the proposed work carries out authentication. CCP uses images i.e., graphical passwords. In CCP, the user has to pick disparate five images rather than choosing click points on the same image. For each image, the user has to choose only one click point. If the user clicks on the image's right position, then the server displays a subsequent image. In CCP, the subsequent image's address is stored on the preceding click point. If the click point is wrong, then the server displays the wrong image

and didn't allow the user for accessing the data. CCP proffers enhanced security and usability. In the verification process, the same SHA-512 algorithm is utilized to transmute the entered username into a hashed value, which also renders more security. Next, this converted Hashed value was contrasted with the previously saved Hash value. If both Hash values were matched and image captcha also matched, then the user is found. Or else, redirect to the Registration form. The verification part has a mathematical denotation of,

$$V = \begin{cases} U(P_v) = P_v & \text{Allowed} \\ U(P_v) \neq P_v & \text{Denied} \end{cases} \quad (3)$$

wherein the term V implies the verification.

Data compression

Subsequent to authentication, the CHA compresses the uploaded data of the approved person. Data compression (DC) method is lessening the file's logical size for saving disk space for easier as well as quicker transmission through a network or the Internet. Data compression decreases data representation duplication, thus increasing usable data volume and helping to minimize text data size and storage, resource use, or transmission power. There are many coding schemes that provide data compression to provide the understanding of communication by users in a distributed environment. In the proposed work, the CHA is employed for every file's types, like text files as well as multimedia files. Huffman coding stands as an entropy encoding algorithm that is employed for lossless DC. The term alludes to the usage of a varied-length codes-table meant for encoding a source symbol (like a character in life) wherein the varied-length code table was derived in a specific way centred on the approximated probability of occurrence intended for every probable value of the source symbol. Centred on the recurring occurrence of the letter, the Binary Trees (BT) is built in the General Huffman algorithm. It encompasses more time to build a tree. In the proposed system, the BT is built centred upon the ASCII code value, which lessens the tree traversing time. For instance, "CLOUD" is a word that is signified as the ASCII code value that is "C-067, L-076, O-079, U-085, and D-068". Grounded upon these ASCII-code values, the BT could be built. These keys were used to track or manage the flow of data and were used to move to and from the data. For ease of recognizing, the code itself was patterned such that most command codes were together and all visual codes were together. The values are generally expressed on the ASCII code columns in decimal, binary, and hexadecimal form. The commencing large number is

built as the parent node, whereas, the parent node's right-side is built with the balance larger numbers and the left side is built with the least numbers. In addition, in the proposed work, the attained last binary values are transmuted into 2s compliments to enhance the security.

Take into account a symbol set $S_s = \{067, 068, 076, 079, 085\}$ with probability set of the symbols on S is P . Here it is taken as a sample set to provide the compression on data with the existing probability using huffman tree and coding. The algorithms are appropriately listed which denotes the symbol wise compression of CHA as follows,

Step 1 A tree having parent node X with probability $P(X)$ below which lies the children 067, 068, 076, 079, and 085 with their respective probabilities.

Step 2 The 3rd and 4th steps are repeatedly done until the list has only one symbol left.

Step 3 The symbols with minimum probability are selected such that Huffman tree has these symbols as children nodes. The parent node is subsequently created with the probability being equal to the total of probability of children nodes.

Step 4 Take away the children node as of X and then assign it below the newly formed sub-parent node.

Step 5 Centred on the path needed to reach the child root as of the parent/sub-parent node, a codeword is allotted.

Step 6 As per every symbol, the encoded string codes are achieved by means of merging the codeword encountered to achieve the symbol as of parent node X .

Step 7 Finally, obtained traverse path of the Huffman tree which is binary value.

Step 8 The 2's complement is taken for the obtained binary value.

Here, the data is compressed. This CHA renders the advantage for the proposed work such as it lessens the tree traversing time and enhances the security level. Subsequent to compression, the compressed data is mathematically written as,

$$C(d)_i = \{C(d)_1, C(d)_2, C(d)_3, \dots, C(d)_n\} \quad (4)$$

where $C(d)_i$ —compressed dataset, $C(d)_n$ —'n'-number of compressed data.

Secure data transfer

After DC, the compressed data is transferred securely to the CS with the utilization of the modified elliptic curve cryptographic (MECC) algorithm. Normally, the possibilities of MITM attacks are provided in the CS. This attack happens while an attacker interrupts communication that are going on between the two parties to eavesdrop in secret. A MITM attack has players like (a) victim, (b)

entity through which the victim is communicating, and (c) “man-in-middle,” who interrupts the victim’s communications. This scenario becomes critical that a victim is not aware of the person who is interrupting (man-in-middle). For averting this attack, this paper utilizes the MECC algorithm.

The mechanism of the ECC algorithm is adopted during the implementation of public-key cryptography. It is chiefly utilized for generating the public key as well as the private key for the purpose of encrypting as well as decrypting the data. The other systems offer security with a 1024-bit key but it renders the same SL with 164-bit key. As it has the capability to render higher SL with battery resources and lower computing power, it is mostly helpful for mobile applications. But the ECC algorithm elevates the probability of implementation errors and hence it is much harder to implement. This would affect the system’s SL. Therefore, for increasing the system’s SL, the proposed work utilizes MECC. In MECC, another key termed secret key is generated. The MECC’ executes encryption and even decryption utilizing private, public, and secret keys.

The proposed algorithm creates a different key to retrieve the system for both administrators and users. Through using the comparable MECC algorithm, the data is encrypted and decrypted. Whenever participants and other administrators want to access data, their identification is authenticated. The applicants are provided with credentials after effective verification. To decrypt the data with all these parameters, the recipients activate the MECC algorithm and produce a secret key. This provides a high level of encapsulation of data.

MECC technique is centered on a curve with certain base points using a prime number function and it is utilized as a maximal limit. The ECC is mathematically evaluated as,

$$p^2 = q^3 + uq + v \quad (5)$$

where u and v signifies the integers.

During cryptographic implementation, the encryption technique’s strength is contingent merely on the mechanism deployed for the key generation. In the proposed system, there are three sorts of keys namely public α_k , private β_k and secret γ_k keys which have to be generated. Primarily, generate α_k as of the server and encrypt it. Secondly, generate β_k on the server-side and decrypt the considered message. Thirdly, generate γ_k as of α_k , β_k , and point on the curve (p_c). During formulation, γ_k is added to the encryption and subtracted as of the decryption. The β_k is picked as of the “n” number of values arbitrarily and α_k is generated grounded on the β_k and p_c .

The Eq. (6) generates α_k .

$$\alpha_k = \beta_k \times p_c$$

Secret Key:

Now, γ_k is developed by evaluating the total of α_k , β_k and p_c . The equation for creating γ_k is evinced below.

$$\gamma_k = \alpha_k + \beta_k + p_c \quad (7)$$

Encryption:

In an encryption phase, original data O_d is transformed into affine point on the curve. Subsequently, the data that are acquired are encrypted. The encrypted information comprises two cipher texts which are mathematically evaluated as,

$$C(t)_1 = (K \times p_c) + \gamma_k \quad (8)$$

$$C(t)_2 = (O_d + (K \times \alpha_k)) + \gamma_k \quad (9)$$

Here, $C(t)_1$ and $C(t)_2$ -two cipher texts, K -random number generated between 1 to $n - 1$, and O_d -original data. Lastly, the encrypted data is uploaded to the CS securely.

On the receiver side, the receiver securely downloads the ciphertext data as of the CS and decrypts the ciphertext utilizing the same MECC algorithm. The decryption function is evaluated as,

$$O_d = (((C(t)_2 - \beta_k) \times C(t)_1) - \gamma_k) \quad (10)$$

Here, γ_k is subtracted as of the message and attained O_d . The proposed MECC could be detailed using the pseudocode evinced in Fig. 3.

This decrypts the data utilizing the MECC algorithm’s decryption function (Eq. 10) and stores them in the individual folders of the uses. The MECC algorithm has a comparatively better computational cost and speed. It also exploits the good exchange data providing another mark to the security.

Results and discussion

This section encompasses a summary of the proposed system’s performance. The proposed system is employed in JAVA. The JAVA platform’s hardware configuration encompasses Intel i5/core i7 processor, 4 GB RAM, as well as 3.20 GHZ CPU speed. Dropbox Cloud Server is utilized for the cloud environment for saving the user data. For a performance assessment, the proposed system takes a video (.mp4) file as an inputted data. The proposed system’s performance is weighed against the prevailing method, which is elucidated in the subsection.

Input: Compressed data $C(d)_i = \{C(d)_1, C(d)_2, C(d)_3, \dots, C(d)_n\}$

Output: Secure transfer

Begin

Generate Public Key, Private Key, Secret Key

for each data **do**

 //Encryption

$$C(t)_1 := ((K := 1, 2, 3, \dots, n-1) * p_c) + \gamma_k$$

$$C(t)_2 := (O_d + (K := (1, 2, 3, \dots, n-1) * \alpha_k)) + \gamma_k$$

 //Decryption

$$O_d = (((C(t)_2 - \beta_k) * C(t)_1) - \gamma_k)$$

end for

End

Fig. 3 Pseudocode for the proposed MECC algorithm

Performance analysis based on compression

The performance is assessed with regards to memory usage (MU) on data uploading, MU on data downloading, data uploading time, data downloading time, encryption time (ET), decryption time (DT), compression ratio (CR), compression time, data size, MU on encryption, MU on decryption as well as SL of the system. The outcomes, therefore, attained are weighed against the prevailing system, all of the terms are expressed as,

- (a) *Memory usage on data uploading* This estimates the total utilized memory of CPU amid the data uploading. The MU on data uploading (M_u) is signified as,

$$M_u = M_{be} - M_{fe} \quad (11)$$

where M_{be} —the total memory prior to encryption and M_{fe} —the free memory space available subsequent to encryption.

- (b) *Memory usage on data downloading* This evaluates the total utilized memory of CPU amid data downloading. The MU on data downloading (M_d) is as

$$M_d = M_{bd} - M_{fd} \quad (12)$$

wherein M_{bd} —the total memory prior to decryption and M_{fd} —the total memory space available subsequent to decryption.

- (c) *Data uploading time* The time taken to gauge the uploading time is computed as,

$$U_t = U_2 - U_1 \quad (13)$$

wherein U_t —the upload time, U_2 —the time subsequent to encryption and U_1 —the time p to encryption.

- (d) *Data downloading time* The downloading time of the data is computed as

$$D_t = D_2 - D_1 \quad (14)$$

wherein D_t —the download time, D_2 —the time subsequent to data decryption and D_1 —the time prior to decryption.

- (e) *Encryption time* The ET is the time that an encryption algorithm encompasses to generates a cipher text as of a plain text. ET is stated as the difference betwixt the encryption ending time and encryption starting time. ET of the system is expressed as,

$$E_{(t)} = E_{\text{end}(t)} - E_{\text{start}(t)} \quad (15)$$

wherein $E_{(t)}$ signifies the encryption time, $E_{\text{end}(t)}$ implies the encryption ending time, and $E_{\text{start}(t)}$ represents the encryption starting time.

- (f) *Decryption time* DT is stated as the difference betwixt the decryption ending time and decryption starting time. The system's DT is mathematically expressed as,

$$D_{(t)} = D_{\text{end}(t)} - D_{\text{start}(t)} \quad (16)$$

wherein $D_{(t)}$ implies decryption time, $D_{\text{end}(t)}$ implies decryption ending time and $D_{\text{start}(t)}$ denotes the decryption starting time.

- (g) *Security level* SL of the existing together with the proposed work is analyzed. Security is extremely significant for cloud storage. The SL is stated as the hacked data divided by means of the number of the actual text. The system's SL is mathematically written as,

$$S_e = H_d / N(O_d) \quad (17)$$

wherein S_e implies the security level, H_d implies a hacked text, and $N(O_d)$ signifies the number of original data.

- (h) *Compression ratio* This is stated as the compressed data's size by the original data's size, which is mathematically written as,

$$C_r = \frac{C(d)_i}{O_d} \quad (18)$$

wherein C_r implies the compression ratio.

- (i) *Compression time* This is stated as the difference betwixt the compression ending time and compression starting time. The system's Compression time is mathematically written as,

$$C_{(t)} = C_{\text{end}(t)} - C_{\text{start}(t)} \quad (19)$$

where $C_{(t)}$ signifies decryption time, $C_{\text{end}(t)}$ implies decryption ending time and $C_{\text{start}(t)}$ indicates the decryption starting time.

- (j) *Memory usage on encryption* It is stated as the difference betwixt the MU after encryption and MU before encryption, which is mathematically expressed as,

$$M(u)_{\text{enc}} = A_e - B_e \quad (20)$$

wherein $M(u)_{\text{enc}}$ implies the MU on encryption, A_e signifies the MU after encryption, and B_e indicates the MU on before encryption.

- (k) *Memory usage on decryption* It is stated as the difference betwixt the MU after decryption (A_d) and MU before decryption (B_d), which mathematically expressed as,

$$M(u)_{\text{enc}} = A_d - B_d \quad (21)$$

where $M(u)_{\text{dec}}$ signifies the MU on decryption.

- (l) *Data size* This is the comparison of the data size before compression and after compression, which is signified as,

$$F_s = O_d (V_s) (A(O_d)) \quad (22)$$

Where F_s implies the data size, V_s signifies the verus, $A(O_d)$ indicates the after compressed data.

The proposed CHA's performance is contrasted with the existing Huffman, which is exhibited in Fig. 4,

Discussion Fig. 4 exhibits the proposed CHA's performance with that of the Huffman centred upon the (a) compression ratio, (b) compression time along with (c) data size. The performance is analysed grounded upon the data size. The data size ranges as of 5–25 MB. After the data size is five MB, the proposed scheme takes 29854 ms time for compression of the data and the prevailing one takes 32016 ms time. Here, the proposed work obtains lesser time than the prevailing technique. Additionally, the compression ratio is higher for the proposed work. After that, the data size is as well compared subsequent to encryption. While the data size is five MB, the proposed system's compressed data size is 2.769765 MB; however, the Huffman compression algorithms condensed data size is 3.539529 MB data. Likewise, for the balance data size, the proposed work attains enhanced performance. Therefore, it is established that the proposed work executed well compared to the prevailing technique.

Discussion Table 1 illustrates the performance comparison of the proposed MECC with the existing ECC concerning ET, DT, along with SL. In Table 1a, the encryption along with DT occupied by these techniques is revealed. The time taken by diverse techniques to execute encryption along with decryption for a disparate number of data sizes is contrasted. For 5 MB data, the prevailing ECC takes 812 ms for encrypting the data, while the proposed MECC takes simply 502 ms for encrypting the data. The ET along with DT for MECC augment slowly, but contrasting with prevailing methods, the proposed one obtains lesser time for encryption along with decryption for every data sizes. In Table 1b, the SL of MECC along with ECC is contrasted. The proposed MECC provides high security (96%) than the ECC (86%). The illustrative demonstration of Table 1 is exhibited in Fig. 5.

Discussion Fig. 5 contrasted the proposed MECC's performance with that of the ECC concerning (a) ET, (b) DT along with (c) SL. The SL of the MECC encompasses 96%, however, the existing is 89%, which is low than the proposed techniques. This SL comparison implies the proposed algorithm's quality level. The ET and DT vary centered on the data size. The data size ranges vary as of 5–25 MB. For

Fig. 4 Compared the performance of the proposed CHA with the existing Huffman in terms of **a** CR, **b** compression time and **c** data size

(a)

(b)

(c)

Table 1 Performance comparison for proposed MECC with existing ECC in terms of (a) ET and DT and (b) SL

(a) Data size (Mb)	Encryption time (ms)		Decryption time (ms)	
	Existing ECC	Proposed MECC	Existing ECC	Proposed MECC
5	812	502	987	821
10	1665	1312	1552	1123
15	2542	2147	2411	2114
20	3122	2689	3244	2885
25	4688	4122	4588	4122
(b) Techniques				Security level (%)
Existing ECC				89
Proposed MECC				96

25 MB data, the proposed work takes 4122 ms for encryption together with decryption, which is 11% lower compared to the prevailing technique. For the entire disparate data sizes, the proposed technique encompasses less time than the prevailing approach. As of this comparison, it can be stated that the MECC attains better performance than the prevailing methodology.

Discussion Fig. 6 illustrates the proposed scheme's performance centred upon (1) data uploading time, (2) data downloading time, (3) MU on data uploading along with (4) MU on data download. In this investigational assessment, the performance of the data uploading time, downloading time, together with MUs are analysed. When the data size is 25 MB, the proposed work has 23441 ms time for uploading the data, 21446 ms time engaged for downloading the data, 761447 KB MU on data uploading along with 864785 KB MU on data download. Likewise, for the 5 MB, 10 MB, 15 MB along with 20 MB, the proposed method attains improved performance. As of the figure scrutiny, it is established that the proposed work offers improved performance.

Discussion Fig. 7 exhibits the MU of the proposed along with existing techniques for encryption together with decryption. For encrypting 5 MB data, the proposed MECC encompasses 256,306 KB of memory while the ECC encompasses 325549 KB of memory, which is high than MECC. Similarly, for decrypting the same 5 MB data, the MECC utilizes 287745 KB of memory along with that the ECC utilizes 334457 KB of memory, which is also high than MECC. When the data size augments, the MU of the proposed as well as existing techniques also augments. However, the proposed MECC takes low memory than ECC for all sizes of data. The proposed works' outcomes are examined by contrasting them with prevailing algorithms. The proposed MECC techniques take lesser time for encryption along with decryption. The proposed MECC method proffers higher levels of security (96%).

Conclusion

As users outsource their sensitive data to cloud providers, Data security, as well as access control, is becoming the challenging ongoing research work in CC. This paper proposed a Modified ECC-based secure data transfer and three-factor authentication system in the untrusted cloud environment to improve SL in the CC environment. The proposed system comprises steps like authentication, data compression, and secure data transfer. During authentication, the SHA-512 algorithm and CCP are utilized. The user uploaded data is then compressed utilizing CHA. Then, the data are securely uploaded to the CS utilizing MECC. The performance is analyzed grounded on data size. The proposed MECC security algorithm and the existing ECC algorithm are compared grounded on their performance in respect of ET, DT, and SL, MU on encryption, and MU on decryption. Then, the proposed CHA and the existing Huffman algorithm are compared and grounded on their performance in respect of CR, data size, and compression time metrics. From these comparisons, the proposed system is found to render excellent performance in considering the existing ones. The proposed MECC attained 96% SL, which is 7% superior to the existing ECC. Thereby, the proposed system has provided better SL than state-of-art approaches. On account of this high security, the proposed approach is commonly utilized in various real-time applications like secure network communication, medical big data transmission, cybersecurity, etc.

In the future, the proposed work can be expanded to store the data in the distributed CS for attaining more security. Presently, a single CS system can fail if the attacker finds the data location. But in the case of distributed CS, the data will be stored in disparate CS, which will fulfill the needs, like data privacy and security.

Fig. 5 Comparative analysis for proposed MECC with the existing ECC **a** encryption time, **b** decryption time and **c** SL

(a)

(b)

(c)

Fig. 6 Performance analysis based on **a** data uploading time, **b** data downloading time, **c** MU on data uploading and **d** MU on data downloading

Fig. 7 Comparative analysis for the security algorithms based on **a** MU on encryption and **b** MU on decryption

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflicts of interest to report regarding the research study.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Zhang Z-H, Xue-Feng J, Li J-J, Jiang W (2012) An identity-based authentication scheme in cloud computing. In: International conference on industrial control and electronics engineering (ICICEE), IEEE, pp 984–986
- Zhang Q, Cheng L, Boutaba R (2010) Cloud computing: state-of-the-art and research challenges. *J Internet Serv Appl* 1(1):7–18
- Mithila S, Kumar PP (2012) Data security through confidentiality in cloud computing environment. *Int J Comput Sci Inf Technol* 2(5):136–840
- Semwal P, Sharma MK (2017) Comparative study of different cryptographic algorithms for data security in cloud computing. In: 3rd International conference on advances in computing, communication and automation (ICACCA) (Fall), IEEE, pp 1–7
- Kumar PR, Raj PH, Jelciana P (2018) Exploring data security issues and solutions in cloud computing. *Proc Comput Sci* 125:691–697
- Singh A, Chatterjee K (2017) Cloud security issues and challenges: a survey. *J Netw Comput Appl* 79:88–115
- Srinivas J, Das AK, Kumar N, Rodrigues J (2018) Cloud centric authentication for wearable healthcare monitoring system. *IEEE Trans Dependable Secure Comput* 17:1–1
- Gupta BB, Yamaguchi S, Chelliah PR (2016) Guest editorial special issues on security trends in mobile cloud computing, web, and social networking. *EURASIP J Inf Secur* 2016:1
- Vyas J, Modi P (2017) Providing confidentiality and integrity on data stored in cloud storage by hash and meta-data approach. *Int J Adv Res Eng Sci Technol* 4(5):38–50
- Ramachandra G, Iftikhar M, Khan FA (2017) A comprehensive survey on security in cloud computing. *Proc Comput Sci* 110:465–472
- Abbas H, Maennel O, Assar A (2017) Security and privacy issues in cloud computing. *Ann Telecommun* 72:233–235
- Sim LJH, Ren SQ, Keoh SL, Aung KMM (2016) A cloud authentication protocol using one-time pad. In: Region 10 conference (TENCON), IEEE, pp 2513–2516
- Hong S, Park S, Park LW, Jeon M, Chang H (2017) An analysis of security systems for electronic information for establishing secure internet of things environments: focusing on research trends in the security field in South Korea. *Future Gener Comput Syst* 82:769–782
- Caviglione L, Podolski M, Mazurczyk W, Ianigro M (2016) Covert channels in personal cloud storage services: the case of dropbox. *IEEE Trans Ind Inf* 13(4):1921–1931
- Halabi T, Bellaiche M (2018) A broker-based framework for standardization and management of cloud security-SLAs. *Comput Secur* 75:59–71
- Jeong Y-S, Park JS, Park JH (2015) An efficient authentication system of smart device using multi factors in mobile cloud service architecture. *Int J Commun Syst* 28(4):659–674
- Pei W, Li J, Li H, Gao H, Wang P (2017) ASCAA: API-level security certification of android applications. *IET Softw* 11(2):55–63
- Hussain SA, Fatima M, Saeed A, Raza I, Shahzad RK (2017) Multilevel classification of security concerns in cloud computing. *Appl Comput Inform* 13(1):57–65
- Kumar MR, Srinivasu N, Reddy LC (2018) Dynamic and secure authentication using IKC for distributed cloud environment. In: Satapathy S, Tavares J, Bhateja V, Mohanty J (eds) Information and decision sciences. *Advances in Intelligent Systems and Computing*, vol 701. Springer, Singapore, pp 107–117. https://doi.org/10.1007/978-981-10-7563-6_12
- Roy A, Dasgupta D (2018) A fuzzy decision support system for multifactor authentication. *Soft Comput* 22(12):3959–3981
- Pitchai R, Jayashri S, Raja J (2016) Searchable encrypted data file sharing method using public cloud service for secure storage in cloud computing. *Wirel Pers Commun* 90(2):947–960
- Rani DR, Geethakumari G (2020) Secure data transmission and detection of anti-forensic attacks in cloud environment using MECC and DLMNN. *Comput Commun* 150:799–810
- Shen J, Deng X, Zhenwu Xu (2019) Multi-security-level cloud storage system based on improved proxy re-encryption. *EURASIP J Wirel Commun Netw* 1:1–12
- Maram B, Gnanasekar JM, Manogaran G, Balaanand M (2018) Intelligent security algorithm for UNICODE data privacy and security in IOT. *SOCA* 13(1):3–15. <https://doi.org/10.1007/s11761-018-0249>
- Namasudra S, Devi D, Kadry S, Sundarasekar R, Shanthini A (2020) Towards DNA based data security in the cloud computing environment. *Comput Commun* 151:539–547
- Li X, Wu F, Kumar N (2018) A secure authentication scheme based on elliptic curve cryptography for IoT and cloud servers. *J Supercomput* 74(12):6428–6453

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.